Reconstrucción 3D densa de escenas utilizando una cámara monocular

Daniel Cores and Manuel Mucientes

Centro Singular de Investigación en Tecnoloxías da Información (CiTIUS),
Universidade de Santiago de Compostela, Santiago de Compostela, España,
daniel.cores@usc.es, manuel.mucientes@usc.es

Resumen La reconstrucción 3D densa de escenas es de gran interés tanto para la navegación de robots como para el modelado 3D de objetos o la realidad aumentada. En este artículo se describe la arquitectura de un sistema capaz de generar una reconstrucción 3D densa del entorno utilizando una cámara monocular. Para ello se ha implementado un algoritmo de estéreo basado en movimiento capaz de calcular un mapa de profundidad en cada imagen para su posterior integración en un mapa denso. La utilización de una cámara monocular permite evitar las desventajas en cuanto al rango y las condiciones de funcionamiento de otros tipos de sensores como las cámaras RGB-D o los pares estéreo. El sistema propuesto ha sido validado tanto en conjuntos de datos sintéticos en escenas interiores como en entornos reales exteriores.

Key words: reconstrucción 3D densa, estéreo con cámara monocular

1. Introducción

La posibilidad de obtener mapas 3D densos mediante una cámara monocular en lugar de otros dispositivos de mayor peso y consumo de potencia, como un sensor LIDAR, puede tener grandes beneficios en determinados tipos de robots como UAVs de reducidas dimensiones. Otros tipos de dispositivos como las cámaras RGB-D no son capaces de ofrecer medidas fiables de profundidad en zonas con iluminación solar, además de tener un rango de medida muy limitado, haciendo inviable su uso en exteriores.

En este trabajo se describe la arquitectura de un sistema capaz de generar una reconstrucción 3D densa del entorno utilizando una cámara monocular. El sistema aplica un algoritmo de SLAM para obtener las posiciones de la cámara y, a partir de ellas, se ha implementado un método de estimación de la profundidad —denominado *Plane Sweep*— que realiza el estéreo entre la imagen actual y una de las imágenes anteriores. Esta imagen se selecciona de tal forma que se maximice la distancia entre las correspondientes posiciones de la cámara manteniendo las imágenes lo más similares posibles. El algoritmo incluye un Filtro de Kalman Extendido (EKF) para la propagación de la profundidad entre imágenes sucesivas. Por último, se aplican una serie de filtros de ruido y se integran los

mapas de profundidad en el modelo 3D final utilizando TSDF (*Truncated Signed Distance Function*).

Este trabajo se basa en [1] con las siguientes diferencias: (i) como algoritmo para el cálculo de las posiciones de la cámara se ha seleccionado ORB-SLAM [2]; (ii) En el método de selección de *keyframes* se utiliza una representación de tipo Bag of Words para comparar las imágenes y poder evaluar la similitud; (iii) Se define un algoritmo de asociación de píxeles entre diferentes mapas de profundidad, útil a la hora de propagar la profundidad entre imágenes sucesivas o comprobar si varias medidas de la misma zona de la escena son consistentes en el tiempo.

El presente documento se estructura de la siguiente forma: en la sección 2 se presenta el trabajo relacionado. En la sección 3 se describe el sistema de reconstrucción 3D densa. En la sección 4 se muestran los resultados obtenidos en diferentes entornos. La sección 5 recoge las conclusiones.

2. Trabajo relacionado

Uno de los primeros trabajos en obtener una reconstrucción 3D densa, precisa y eficiente fue KinectFusion [3], un sistema que utiliza los mapas de profundidad generados por una cámara RGB-D para la construcción de un modelo 3D de gran calidad representado mediante TSDF (*Truncated Signed Distance Function*). La principal desventaja de KinectFusion radica en que se define un cubo de tamaño fijo (para alcanzar tasas de actualización del modelo en tiempo real) dentro del cual se puede realizar la reconstrucción. Sobre este trabajo han surgido nuevas contribuciones que intentan mitigar este problema mediante la utilización de un cubo móvil, como en el caso de [4], en el que simplemente se descartan los datos que se dejan atrás y se mapea el contenido del volumen a la nueva posición del mismo cada vez que es necesario moverlo, permitiendo localizar la cámara en largas trayectorias ya que se mantiene en todo momento un modelo del entorno cercano. Una alternativa que sí utiliza la información de superficie que se deja atrás la encontramos en [5], donde se crea un mapa global que se actualiza a medida que se visitan diferentes zonas del entorno. No se define ninguna estrategia que permita reutilizar datos de zonas ya visitadas y que han salido del cubo de reconstrucción, provocando la aparición de superposición de superficies en algunas circunstancias.

La mayor parte de los problemas asociados a estas aproximaciones se originan en la utilización de una estructura de datos poco eficiente, un cubo de tamaño fijo en el que la mayor parte de los *voxels* se encuentran sin información, ya que se trunca la distancia máxima hasta la cual se almacena un valor. En [6] se propone una nueva aproximación basada en la utilización de una *Tabla Hash* compuesta de pequeñas agrupaciones de *voxels*, de este modo se optimiza el espacio necesario para almacenar el mapa al reservar memoria únicamente para aquellas zonas de la escena que contienen información. En CHISEL [7] se hace una reimplementación de esta idea capaz de ejecutarse en tiempo real en

dispositivos móviles a costa de una menor resolución pasando de unos 4mm a unos 2cm.

Un componente clave en este tipo de sistemas es la localización de la cámara en el entorno. En los sistemas basados en cámaras RGB-D como [3] esta posición se utiliza para integrar los nuevos mapas de profundidad en el lugar adecuado del mapa. En las implementaciones que introducen la noción de cubo móvil [4, 5], la localización de la cámara también es necesaria para mover el cubo de forma correcta y actualizar el modelo en consecuencia. En todos estos casos se utilizan variantes del algoritmo ICP, que obtiene la posición de la cámara realizando una minimización de la distancia entre los puntos 3D del mapa de profundidad y las zonas correspondientes del modelo 3D.

Al utilizar algoritmos de estéreo basados en la selección de *keyframe* para la obtención de los mapas de profundidad, el cálculo de la transformación de la cámara cobra mayor importancia ya que determina en gran medida la calidad de los mapas generados, además de la integración en el modelo. En este tipo de sistemas se necesita conocer la transformación de la cámara antes de extraer el mapa de profundidad con lo que una aproximación como la anterior no es válida. En [8] se utilizan características FAST para la triangulación de ciertos puntos creando un mapa disperso que ayuda a la localización de la cámara. Este mapa disperso es diferente del modelo 3D generado y se debe crear y mantener en paralelo. En [1] se pone de manifiesto que las técnicas básicas de *odometría visual* combinadas con unidades de medición inercial (IMU) [9] no aportan la calidad suficiente en los cálculos de las transformaciones a corto plazo.

3. Sistema de reconstrucción 3D densa

En la figura 1 se muestra el esquema del sistema de reconstrucción 3D densa. Para determinar la posición de la cámara a lo largo del recorrido se utilizará el algoritmo ORB-SLAM [2]. Además, para la estimación de la profundidad se debe seleccionar un *keyframe* lo suficientemente similar a la imagen actual, pero con una transformación en la posición de la cámara lo suficientemente grande para poder realizar la triangulación. Para llevar a cabo esta tarea, en este trabajo se utiliza una técnica similar a la propuesta en [2] para el cierre de lazos, en donde para cada imagen se extraen las características ORB [10] lo más dispersas posibles. De este modo, se evita que las características extraídas en una imagen se concentren en una zona provocando que, potencialmente, se seleccione un *keyframe* que comparta un gran número de puntos característicos con la imagen actual pero en una pequeña región de la misma, ofreciendo un pobre rendimiento a la hora de calcular el mapa de profundidad completo. Una vez echo esto, se añade la nueva imagen a una base de datos de tipo *Bag of Words* (BoW) [11].

Todas las nuevas imágenes se añaden como posibles *keyframes* y, al contrario de lo que sucede en [2], únicamente se eliminan imágenes antiguas (por encima de un umbral) de la base de imágenes. De este modo, se previene que el número de posibles *keyframes* crezca demasiado, sobre todo en largos recorridos en exteriores. Por otro lado, es más probable que dos imágenes cercanas contengan

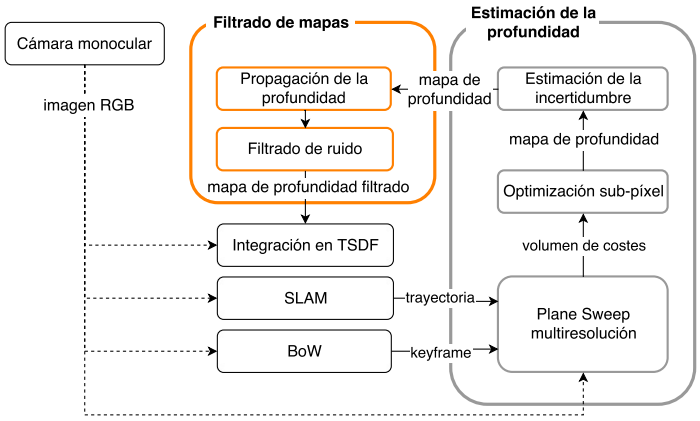

Figura 1. Arquitectura del sistema.

información sobre la misma región. Para el cálculo de la puntuación se utiliza un vocabulario previamente generado. A cada punto característico de la imagen se le asocia una palabra creando un vector $v_t \in \mathbb{R}^W$, siendo W el tamaño del vocabulario. Para cada palabra en el vector se le asigna una ponderación de tipo *tf-idf* (*term frequency-inverse document frequency*). Para comparar las representaciones de dos imágenes (v_1 y v_2) se utiliza la siguiente ecuación [11]:

$$s(v_1, v_2) = 1 - \frac{1}{2} \left| \frac{v_1}{|v_1|} - \frac{v_2}{|v_2|} \right| \tag{1}$$

En las primeras posiciones del *ranking* devuelto por BoW aparecen imágenes con gran parecido a la imagen actual, pero en la mayor parte de los casos, no se encuentran a la distancia suficiente para realizar la triangulación. Por este motivo, en lugar de seleccionar la primera imagen, se escoge aquella cuya posición de la cámara se encuentre los más separada posible de la posición actual y cuya puntuación de similitud se encuentra por encima de la puntuación de la primera corregida por un factor (puntuación mínima aceptable). De este modo se añade un compromiso entre similitud y distancia entre imágenes. Con el objetivo de prevenir que el mismo *keyframe* se escoja un alto número de veces, se selecciona de forma aleatoria uno de los tres *keyframes* más alejados que se encuentran dentro del umbral de similitud. De este modo, si algún *keyframe* introduce medidas erróneas, se minimiza el posible efecto en el resultado final.

3.1. Estimación de la profundidad

Una vez seleccionado el *keyframe* y calculada la transformación entre ambas imágenes se ejecuta el algoritmo de estéreo *Plane Sweep* [12] utilizando ZNCC (*Zero-mean Normalized Cross Correlation*) para el cálculo de la función de coste. Los planos se disponen utilizando una distancia fija y en paralelo al plano de la

imagen en el *frame* actual (Figura 2). Se realiza este proceso para dos niveles diferentes de resolución manteniendo el mismo tamaño de ventana para ZNCC, y se agrega el resultado obteniendo un único volumen de costes final [13]. En lugar de utilizar los valores de profundidad directamente, se trabaja con la inversa de la profundidad. De este modo, se puede asumir una distribución normal en la incertidumbre, necesario para la aplicación del EKF a la hora de propagar la hipótesis de profundidad en cada píxel.

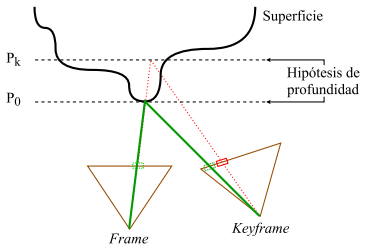

Figura 2. Para el píxel de la imagen de referencia (cámara de la izquierda) se prueban diferentes hipótesis de profundidad (por ejemplo, los planos P_0 y P_k). La proyección en el *keyframe* a través del plano P_0 se corresponde con una zona de la imagen que representa la misma zona de la escena con lo que a la profundidad asociada al píxel se le asigna la distancia a la que se ha establecido P_0. Por contra, el plano P_k da una proyección incorrecta, lo que genera un peor valor de ZNCC.

3.2. Optimización sub-píxel

Con el fin de obtener una precisión mayor a la ofrecida por el conjunto de planos seleccionado, se ajusta una parábola que pasa por el mínimo de la función de coste y por los dos puntos que rodean dicho mínimo. El valor final de profundidad vendrá determinado por el mínimo de la parábola (μ_0 en la figura 3). Todos los píxeles cuyo mínimo de la función de coste esté por encima de un umbral serán descartados y se considerará la medida como desconocida [1].

3.3. Estimación de la incertidumbre

Para la actualización del EKF es necesario estimar la varianza en cada píxel. Se define la desviación estándar como (Figura 3):

$$\sigma_0 = max(\mu_0 - d_{min}^{inv}, d_{max}^{inv} - \mu_0)$$

siendo μ_0 la inversa de la profundidad y d_{min}^{inv} y d_{max}^{inv} los valores mínimo y máximo de profundidad a partir de los cuales la función de coste toma valores por encima de γc_{min} siendo c_{min} el coste asociado al mínimo de la parábola y γ un parámetro del sistema [1].

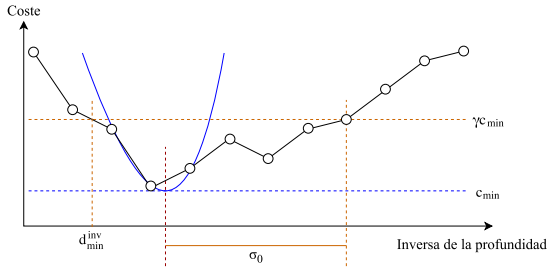

Figura 3. Optimización de la función de coste y estimación de la desviación típica.

3.4. Propagación de la profundidad

Con el fin de realizar la integración a lo largo del tiempo de los mapas de profundidad, el sistema utiliza un EKF [1]. Para la predicción de la inversa de la profundidad en el *frame* i ($\overline{\mu}_i$) y la varianza asociada ($\overline{\sigma}_i^2$) se utilizan la inversa de la profundidad y la varianza en el *frame* anterior (μ_{i-1} y σ_{i-1}^2) así como la transformación de la cámara en el eje óptico (t_z) y la incertidumbre asociada a esta transformación ($\sigma_{t_z}^2$) que se estima como una proporción de la cantidad de movimiento:

$$\overline{\mu}_i = \left(\mu_{i-1}^{-1} - t_z\right)^{-1} \tag{2}$$

$$\overline{\sigma}_i^2 = \left(\frac{\overline{\mu}_i}{\mu_{i-1}}\right)^4 \sigma_{i-1}^2 + \overline{\mu}_i^4 \sigma_{t_z}^2 \tag{3}$$

Las estimaciones anteriores se corrigen mediante la inversa de la profundidad (μ_0) y la varianza (σ_0), calculadas en las etapas descritas anteriormente.

$$\mu_i = \frac{\overline{\sigma}_i^2 \mu_0 + \sigma_0^2 \overline{\mu}_i}{\overline{\sigma}_i^2 + \sigma_0^2} \tag{4}$$

$$\sigma_i^2 = \frac{\overline{\sigma}_i^2 \sigma_0^2}{\overline{\sigma}_i^2 + \sigma_0^2} \tag{5}$$

3.5. Asociación de píxeles entre mapas de profundidad

Una parte fundamental en este proceso consiste en relacionar los píxeles de la imagen anterior con la imagen actual. Para ello se calcula la posición 3D asociada al píxel (x_0, y_0). La representación 3D de cada píxel se obtiene como:

$$z = 1/\mu \tag{6}$$

$$x = ((x_0 - c_x)z)/f_x \tag{7}$$

$$y = ((y_0 - c_y)z)/f_y \tag{8}$$

donde f_x, f_y, c_x y c_y son los parámetros de configuración de la cámara (focal, y punto principal).

Una vez calculada la posición 3D se proyecta en la nueva imagen asociando la medida de profundidad al píxel correspondiente. Esta aproximación puede generar un gran número de píxeles sin inicializar. En la figura 4 se puede ver como al utilizar simplemente las proyecciones de los puntos, en los píxeles del centro no se propagan los valores de profundidad correspondientes ya que ningún punto se proyecta en ellos. Como solución a este problema, en [1] se propone la proyección de una malla de triángulos en lugar de simplemente los puntos, cubriendo una mayor cantidad de píxeles. En este trabajo se ha utilizado una implementación en GPU del algoritmo de triangulación de Delaunay en dos dimensiones [14] sobre los píxeles de la imagen anterior. Esta técnica de triangulación tiende a generar triángulos lo más equiláteros posibles maximizando los ángulos mínimos. Una vez hecho esto, se hacen las proyecciones de los vértices de los triángulos en la imagen actual.

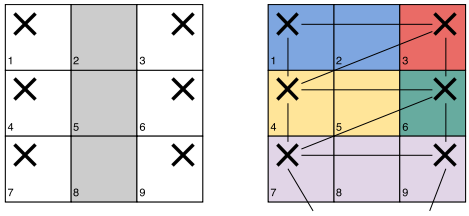

Figura 4. En la figura de la izquierda se realiza una proyección de los puntos 3D (las cruces simbolizan las proyecciones) calculados a partir de la información de profundidad de la imagen anterior, quedando sin ningún valor asignado los píxeles 2, 5 y 8. En la figura de la derecha se utiliza una malla de triángulos que permite cubrir un mayor número de píxeles, asociando cada píxel con los vértices del triángulo al que pertenece su centro (píxeles con el mismo color pertenecen al mismo triángulo).

Por último, se utiliza un contador de validez [1] que se incrementa (hasta un máximo) para cada medida consistente y se decrementa con cada medida inconsistente. Cuando el contador de validez alcanza cero, el estado asociado del EKF se descarta y se volverá a inicializar con la siguiente medida. Una medida es inconsistente cuando no se cumple:

$$|\overline{\mu}_i - \mu_0| < \overline{\sigma}_i - \sigma_0 \tag{9}$$

Por lo tanto, además de propagar la hipótesis de profundidad y la varianza asociada, también se debe propagar este contador. Para ello se utiliza la información de los tres puntos que forman el triángulo que contiene el centro de cada píxel. Se descartan aquellos triángulos cuya diferencia de profundidad entre sus vértices supere un umbral preestablecido. Además, también se impone un límite en el tamaño máximo de los lados de cada triángulo previniendo que un único triángulo pueda afectar a un gran número de píxeles.

El valor de profundidad del nuevo píxel se calcula a partir de la intersección entre el rayo que pasa por el centro de dicho píxel y el plano que contiene al triángulo correspondiente. Dado un píxel (x, y) así como la normal del plano $(\mathbf{n})$ y uno de los vértices del triángulo (P), la profundidad asociada se calcula como:

$$profundidad = \frac{f_x f_y (P \cdot \mathbf{n})}{f_y n_x (x - c_x) + f_x n_y (y - c_y) + f_x f_y n_z} \tag{10}$$

Para el caso de la varianza, se selecciona el máximo de entre las varianzas asociadas a cada uno de los vértices, mientras que para el contador de validez se utiliza el mínimo de los tres vértices. De este modo, se sigue una estrategia conservadora, ya que una varianza alta puede suponer que el valor se descarte mediante filtros posteriores, mientras que si el contador de validez alcanza cero se descarta el valor de profundidad del píxel directamente.

Por último, se aplica un suavizado por mediana definiendo un tamaño de ventana de 3 píxeles.

3.6. Filtrado de puntos fuera de rango

La presencia de ruido en los mapas de profundidad ocasiona un gran deterioro en la calidad del modelo 3D final cuando se utiliza TSDF para representar la superficie. Por ello, la siguiente etapa del sistema es la aplicación de un conjunto de filtros [1].

Consistencia temporal Se comparan mediciones separadas 0.25 segundos y en caso de que no coincidan se descarta el valor de profundidad. Para realizar las asociaciones de píxeles entre los dos mapas de profundidad se emplea el mismo método basado en la triangulación de Delaunay utilizado para propagar la hipótesis de profundidad.

Varianza máxima Se establece un umbral máximo sobre la varianza a partir del cual se descartan todos los píxeles. Mediante la ecuación 11 se calcula la varianza asociada a cada píxel $\sigma^2_{i,d}$ a partir de la varianza de la inversa de la profundidad σ^2_i:

$$\sigma^2_{i,d} = J_x \sigma^2_i J_x^T = \frac{1}{\mu_i^4} \sigma^2_i \tag{11}$$

Ángulo máximo Se impone un ángulo máximo entre la normal de la superficie en cada punto y la dirección con la que se observa dicho punto.

Análisis de componentes conexas Se filtran pequeñas agrupaciones de puntos que se encuentran aisladas. Para generar las agrupaciones se parte de un píxel que todavía no ha sido procesado y se añaden al grupo actual todos los píxeles vecinos cuya diferencia de profundidad con el actual se encuentre por debajo de un umbral. Se realiza el proceso de forma recursiva hasta que no se añadan más vecinos al grupo. A continuación, se selecciona un nuevo píxel no procesado, y se genera un nuevo grupo, repitiendo el proceso hasta que no queden más píxeles sin procesar.

3.7. Integración en el modelo 3D

Una vez obtenido el mapa de profundidad final se integra en un modelo 3D representado mediante un TSDF (*Truncated Signed Distance Function*). Como estructura de datos se utiliza la aproximación híbrida propuesta en [7] en la que se utiliza una *Tabla Hash* para el almacenamiento de pequeñas agrupaciones de *voxels*. De este modo se consigue un sistema con una gran eficiencia en memoria permitiendo el manejo de grandes mapas sin la necesidad de convertir la superficie a otra estructura de datos —como una nube de puntos— como ocurre en [5].

4. Resultados

Para evaluar la precisión y la completitud de los mapas de profundidad generados se ha utilizado el conjunto de datos de prueba ICL-NUIM [15]. Nos basaremos en las definiciones de precisión y completitud descritas en [1], entendiendo la precisión como el porcentaje de píxeles con una medida válida cuya diferencia en profundidad con respecto a la real es inferior a 7.5cm. La completitud se define como el porcentaje de puntos con un error inferior a 7.5cm sobre el total de píxeles de la imagen.

En la tabla 1 se muestran los resultados obtenidos sobre las cuatro trayectorias disponibles en el conjunto de datos para reconstrucción 3D. Se aportan los resultados tanto en la resolución original (640x480) como haciendo una reducción a 320x240 píxeles. La utilización de las imágenes en tamaño original ofrece mejores resultados tanto en precisión como en completitud en todos los casos llegando a mejorar notablemente en el caso de *Living room 1*.

Tabla 1. Resultados obtenidos sobre el conjunto de datos ICL-NUIM.

	QVGA		VGA	
	Precisión	Completitud	Precisión	Completitud
Living room 0	94.4 %	37.1 %	98.0 %	43.0 %
Living room 1	84.1 %	13.6 %	96.2 %	31.6 %
Living room 2	89.0 %	33.2 %	94.3 %	37.9 %
Living room 3	88.6 %	29.6 %	94.7 %	32.2 %

En la Figura 5 se puede ver la mejora de la precisión, a medida que se aplican los diferentes componentes del sistema, para cada una de las trayectorias, tanto en VGA como en QVGA. Se observa que, en general, la consistencia temporal y el análisis de componentes conexas suponen una mejora importante en todos los casos. Por otro lado, el suavizado por mediana supone una mejora para las pruebas con mayor resolución, mientras que empeora en los casos en los que se

utiliza QVGA. En la figura 6 se puede ver tanto el mapa de profundidad de partida como el resultado de aplicar todos los filtros, así como una imagen del modelo 3D generado.

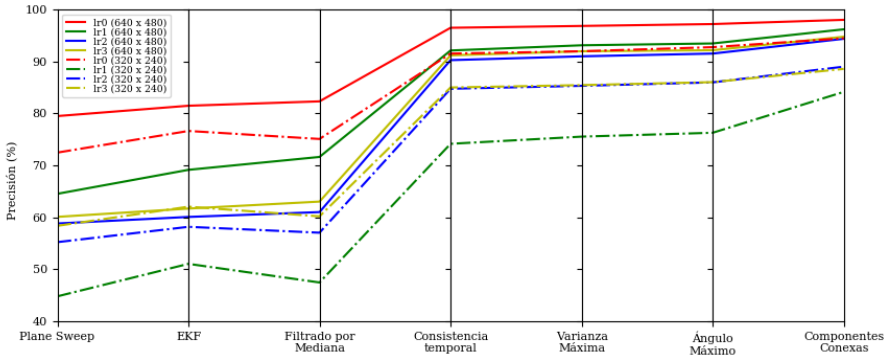

Figura 5. Evolución de la precisión en promedio para todas las imágenes.

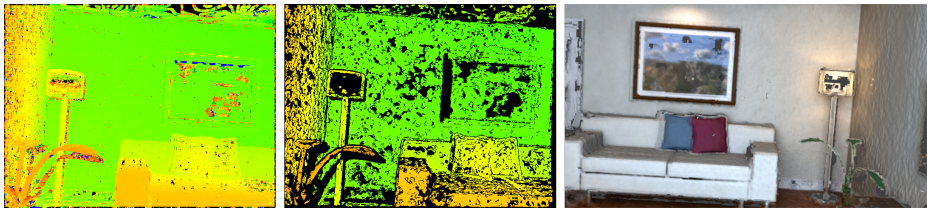

Figura 6. En la imagen de la izquierda se observa el mapa de profundidad calculado a través de *Plane Sweep*. En la siguiente imagen se muestra el mapa de profundidad resultado de aplicar todas las etapas de filtrado. Por último, se representa una imagen del modelo 3D generado.

En las figuras 7 y 8 se muestran reconstrucciones 3D de escenas en exteriores con mapas de gran tamaño con diferentes niveles de iluminación. Estas imágenes se han obtenido con la cámara de un móvil a una resolución de 1280x720 píxeles y se han reducido a 640x360 píxeles para realizar la estimación de la profundidad y la reconstrucción del modelo 3D. Al tratarse de imágenes reales grabadas con una cámara monocular no es posible extraer métricas de precisión y completitud ya que los valores reales no están disponibles.

5. Conclusiones

En este trabajo se ha propuesto un sistema completo de reconstrucción 3D densa utilizando únicamente las imágenes de una cámara monocular. Para la

Figura 7. Edificios bajo iluminación solar directa con sombras (Distancia recorrida: 90m).

Figura 8. Fachada con poca iluminación (Distancia recorrida: 100m).

localización de la cámara, hemos seleccionado ORB-SLAM, un sistema que ofrece grandes resultados tanto en interiores como en exteriores. En la obtención de los mapas de profundidad se ha prestado especial atención a la eliminación de ruido, algo fundamental al no contar con dispositivos activos de medición de la profundidad.

Se ha probado el sistema en diferentes entornos, tanto sintéticos (ICL-NUIM), como en largas trayectorias en exteriores con diferentes condiciones de iluminación. En ambos casos obteniendo buenos resultados consiguiendo reducir el ruido asociado generando modelos densos con altos niveles de completitud.

Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Economía, Industria y Competitividad (TIN2014-56633-C3-1-R) y la Consellería de Cultura, Educación y Ordenación Universitaria de la Xunta de Galicia (GRC2014/030 y acreditación 2016-2019, ED431G/08). Estas subvenciones son cofinanciadas por el Fondo Europeo de Desarrollo Regional (programa ERDF/FEDER).

Referencias

1. Schöps, T., Sattler, T., Häne, C., Pollefeys, M.: Large-scale outdoor 3D reconstruction on a mobile device. Comp. Vision and Image Underst. **157** (2017) 151–166
2. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Trans. on Robotics **31**(5) (2015) 1147–1163
3. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: Real-time dense surface mapping and tracking. In: 10th IEEE Int. Symposium on Mixed and Augmented Reality (ISMAR). (2011) 127–136
4. Roth, H., Vona, M.: Moving volume KinectFusion. In: British Machine Vision Conf.erence (BMVC). Volume 20. (2012) 1–11
5. Whelan, T., Kaess, M., Johannsson, H., Fallon, M., Leonard, J.J., McDonald, J.: Real-time large-scale dense RGB-D SLAM with volumetric fusion. The Int. Journal of Robotics Research **34**(4-5) (2015) 598–626
6. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3D reconstruction at scale using voxel hashing. ACM Trans. on Graphics (TOG) **32**(6) (2013) 169
7. Klingensmith, M., Dryanovski, I., Srinivasa, S., Xiao, J.: Chisel: Real time large scale 3D reconstruction onboard a mobile device using spatially hashed signed distance fields. In: Robotics: Science and Systems. Volume 4. (2015)
8. Pradeep, V., Rhemann, C., Izadi, S., Zach, C., Bleyer, M., Bathiche, S.: Monofusion: Real-time 3D reconstruction of small scenes with a single web camera. In: IEEE Int. Symposium on Mixed and Augmented Reality (ISMAR). (2013) 83–88
9. Hesch, J.A., Kottas, D.G., Bowman, S.L., Roumeliotis, S.I.: Camera-IMU-based localization: Observability analysis and consistency improvement. The Int. Journal of Robotics Research **33**(1) (2014) 182–201
10. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An efficient alternative to SIFT or SURF. In: IEEE Int. Conf. on Computer Vision (ICCV). (2011) 2564–2571
11. Gálvez-López, D., Tardos, J.D.: Bags of binary words for fast place recognition in image sequences. IEEE Trans. on Robotics **28**(5) (2012) 1188–1197
12. Collins, R.T.: A space-sweep approach to true multi-image matching. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). (1996) 358–363
13. Zhang, K., Fang, Y., Min, D., Sun, L., Yang, S., Yan, S., Tian, Q.: Cross-scale cost aggregation for stereo matching. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). (2014) 1590–1597
14. Cao, T.T., Nanjappa, A., Gao, M., Tan, T.S.: A GPU accelerated algorithm for 3D Delaunay triangulation. In: ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. (2014) 47–54
15. Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A benchmark for RGB-D visual odometry, 3D reconstruction and slam. In: IEEE Int. Conf. on Robotics and automation (ICRA). (2014) 1524–1531